User's guide to ddsip.vSD – A C Package for the Dual Decomposition of Stochastic Programs with Dominance Constraints Induced by Mixed-Integer Linear Recourse

U. Gotzes, F. Neise
Department of Mathematics
University of Duisburg-Essen, Campus Duisburg
Lotharstr. 65, D-47048, Germany
gotzes@math.uni-duisburg.de, neise@math.uni-dusburg.de

March 12, 2008

1 Introduction

`ddsip.vSD` is a C-implementation of a number of scenario decomposition algorithms for stochastic linear programs with first- or second-order stochastic dominance constraints induced by mixed-integer linear recourse. The program is based on a previous implementation of scenario decomposition algorithms for mean-risk models of A. Märkert [20]. Main idea of the decomposition algorithms is the Lagrangean relaxation of unavoidable scenario coupling second stage constraints and of (some) nonanticipativity constraints. A branch-and-bound algorithm to reestablish nonanticipativity is employed. The original scenario decomposition algorithm for stochastic linear programs with mixed-integer recourse has been developed in [5]. Extensions including the treatment of mean-risk models have been made in [19]. The algorithms focussing stochastic dominance have been elaborated in [10, 11].
For the dual optimization we use `ConicBundle` – a C++ implementation provided by C. Helmberg, see [12]. We use the `CPLEX` callable library to solve the mixed-integer subproblems in the branch-and-bound tree, see [14]. The current version of `ddsip.vSD` relies on `CPLEX 9.1.3`.
Unfortunately, we did not yet have the chance to debug and optimize the code in a sufficient way. We ask the user to support fixing bugs by reporting them to us as they occur.
This manual describes the format of the input files and the data contained in the output files of `ddsip.vSD`. We try to provide all necessary information on the input parameters.

2 Stochastic programs with dominance constraints induced by mixed-integer linear recourse

`ddsip.vSD` is appropriate to solve *stochastic programs with first- or second-order dominance constraints induced by mixed-integer linear recourse*. Such problems were first investigated by

[11, 10].
Two-stage stochastic programming models are derived from random optimization problems with information constraints. We start out from the following random mixed-integer linear program.

$$\min\{c^\top x + q^\top y \ : \ Tx + Wy = z(\omega),\ x \in X,\ y \in Y\}, \tag{1}$$

together with the information constraint that x must be selected without anticipation of $z(\omega)$. This leads to a two-stage scheme of alternating decision and observation: The decision on x is followed by observing $z(\omega)$ and then y is taken, thus depending on x and $z(\omega)$. Accordingly, x and y are called first- and second-stage decisions, respectively. X and Y are polyhedra, possibly involving integer requirements to some vector components. z is a random vector on some probability space. Note, that also c, q, T and W may have stochastic entries and that this case is also covered by our implementation. Furthermore - besides other implicit regularity assumptions ([17]) - all objects in (1) may have conformal dimensions.
The mentioned two-stage dynamics becomes explicit by the following reformulation of (1)

$$\begin{aligned} \min_x \Big\{ c^\top x \ &+ \ \min_y \{q^\top y \ : \ Wy = z(\omega) - Tx,\ y \in Y\} \ : \ x \in X \Big\} \\ &= \ \min_x \{c^\top x \ + \ \Phi(z(\omega) - Tx) \ : \ x \in X\} \end{aligned} \tag{2}$$

where

$$\Phi(t) := \min\{q^\top y \ : \ Wy = t,\ y \in Y\}. \tag{3}$$

The function Φ, called the value function of the mixed-integer linear program

$$\min\{q^\top y \ : \ Wy = t,\ y \in Y\},$$

has been studied in parametric optimization, [2, 4].
In view of (2), the random optimization problem (1) gives rise to the family of random variables

$$\Big(c^\top x + \Phi(z(\omega) - Tx)\Big)_{x \in X}. \tag{4}$$

Thus every first-stage decision $x \in X$ induces a random variable $f_x(\omega) := c^\top x + \Phi(z(\omega) - Tx)$. Traditional two-stage stochastic programming aims at optimizing nonanticipative decisions, i.e., finding a "best" x, or in other words a "best" member in the family (4) of random variables. For the specification of "best", statistical parameters reflecting mean and/or risk are used. Employing the weighted sum of $\mathbb{E}$ and some risk measure $\mathcal{R}$ leads to mean-risk models (cf. [1, 3, 8, 16, 23, 24, 25])

$$\min\{\mathbb{E}(f_x) \ + \ \rho \cdot \mathcal{R}(f_x) \ : \ x \in X\} \qquad (\rho \geq 0 \text{ fixed}). \tag{5}$$

Here, we take an alternative view. Rather than heading for "best" members of (4), we want to identify "acceptable" members, and optimize over them. This leads to the new class of stochastic integer programs, see (6) below, that we address with our software.
Stochastic dominance, an established concept in decision theory [21, 22], provides a possibility to formalize the above mentioned "acceptability". We deal with first- and second-order stochastic dominance in the case where the involved random variables have only finitely many realizations. Let Y a random variable on some probability space and $\mathsf{Y}(\Omega) = \{y_1, \ldots, y_K\}$. When preferring small outcomes to big ones, a random variable X is said to dominate a

random variable Y to first order ($\mathsf{X} \succeq_1 \mathsf{Y}$) iff $\mathbb{P}(\mathsf{X} \leq y_k) \geq \mathbb{P}(\mathsf{Y} \leq y_k),\ \forall k = 1, \ldots, K$. Second-order dominance ($\mathsf{X} \succeq_2 y_k$) holds, iff $\mathbb{E}((\mathsf{X} - y_k)_+) \leq \mathbb{E}((\mathsf{Y} - y_k)_+),\ \forall k = 1, \ldots, K$, i.e. iff the expected excess of X above each outcome of Y is less than or equal to the expected excess of Y above its realizations ($(.)_+ := \max\{.\,, 0\}$).

Coming back to our two-stage random optimization problem (1) and the related family (4), we assume some (random) benchmark cost profile, i.e. a random variable d be given. We consider only those $x \in X$ "acceptable" for which the corresponding f_x dominates the benchmark profile d stochastically to first or second order, respectively. Over all "acceptable" $x \in X$ we optimize some function $g : X \to \mathbb{R}$. This leads to the following stochastic program with dominance constraint induced by mixed-integer linear recourse

$$\min\{g(x) \ :\ f_x \ \succeq_i \ d,\ x \in X\}\ ,\ i = 1, 2 \tag{6}$$

Note that the case of multiple dominance constraints is also covered: Multiple first order constraints $f_x \ \succeq_1 \ d_i\ ,\ \ i = 1, \ldots, N$ can be expressed through a single dominance constraint $f_x \ \succeq_1 \ d$ by choosing d equal the random variable whose distribution function is the pointwise maximum of the distribution functions of the d_i. For the case of multiple second order constraints we refer the reader to Theorem 3.2 in [18]. In [10] and [11] the authors have shown that (6) can be approximated through discrete approximations of f_x and d if their range is not a finite set by nature. On the other hand (6) can be shown to be equivalent to deterministic block-structured large-scale mixed-integer linear programs in the case of finitely many realizations of f_x and d (cf. [10, 11]). However these so called deterministic equivalents quickly become extraordinary large and can often not be tackled with standard solvers alone such that the application of the described scenario decomposition method might be useful.

Note that there exists a more efficient algorithm for the treatment of first-order models with a linear second stage. This method uses duality theory and is described in [7].

We briefly discuss the implemented algorithm for the second-order case. The treatment of first-order instances is basically the same. Assume a finite number of scenarios, i.e. $z(\Omega) = \{z_1, \ldots, z_L\}$, with corresponding probabilities π_l is given and that g in 6 is linear. Then (cf. [10]), problem (6) turns into

$$\min\left\{ g^\top x : \begin{array}{lll} c^\top x + q^\top y_l - v_{lk} & \leq \ d_k & \forall l\ \forall k \\ Tx + Wy_l & = \ z_l & \forall l \\ \sum_{l=1}^{L} \pi_l v_{lk} & \leq \ \mathbb{E}((d - d_k)_+) & \forall k \\ x \in X,\ y_l \in Y & v_{lk} \geq 0 & \forall l\ \forall k \end{array} \right\} \tag{7}$$

The program (7) is a large-scale deterministic mixed-integer linear program with a block-angular structured constraint matrix. Note that the K-multiplicity of the second-stage variables and of the second-stage constraints as in [10, 11] is not necessary, since if the problems there are solved, chose $y_i = \arg\min_{k=1,\ldots,K}\{q^T y_{ik}\}$ and (7) is also solved. And if (7) is solved choose $y_{ik} = y_i\ ,\ \ \forall k = 1, \ldots, K$ in the deterministic equivalents in [10, 11].

By introducing scenario many copies $x_l\ ,\ \ l = 1, \ldots, L$ of the first-stage variables, an equivalent formulation of (7) is given by

$$\min_{x_l, y_l}\left\{ \sum_{l=1}^{L} \pi_l \cdot g^\top x_l \quad : \quad \begin{array}{l} x_1 = \ldots = x_L\ (*) \\ \sum_{l=1}^{L} \pi_l v_{lk} \leq \mathbb{E}((d - d_k)_+)\ (**) \\ (x_l, y_l) \in M_l,\ \forall l,\ \forall k \end{array} \right\} \tag{8}$$

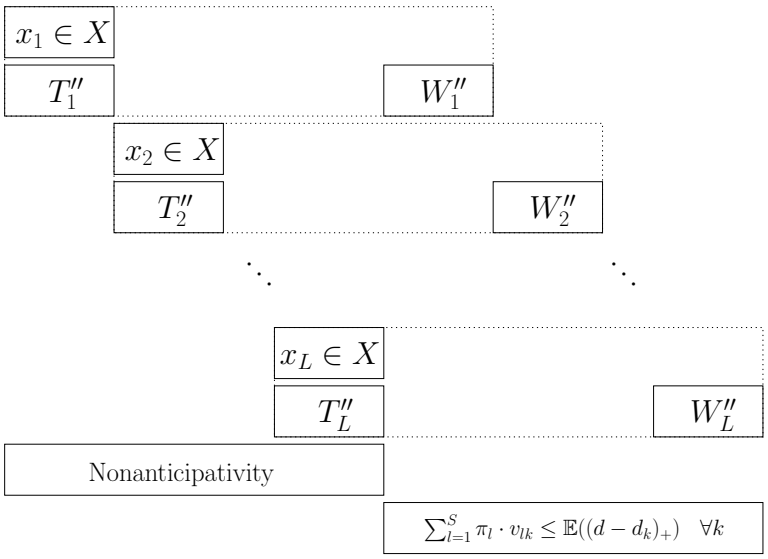

Figure 1: Structure of the constraint matrix

where for $l = 1, \ldots, L$

$$M_l := \left\{ (x_l, y_l) \in X \times Y : \begin{array}{lll} \exists v_{lk} \geq 0 \ , \text{ such that:} & & \\ c^\top x + q^\top y_l - v_{lk} & \leq \ d_k & \forall l \ \forall k \\ Tx + W y_l & = \ z_l & \forall l \ \forall k \end{array} \right\} \tag{9}$$

Considering the constraint matrix of (8), we can identify L single-scenario subproblems solely coupled by the equality constraints $(*)$ (*nonanticipativity*) on the copies of the first-stage variables and the "*dominance-modelling-constraint*" $(**)$ including second-stage variables belonging to different scenarios. The problem decomposes when we relax the nonanticipativity (some/most are omitted, some can be treated with Lagrangean relaxation) and the K constraints

$$\sum_{l=1}^{L} \pi_l v_{lk} \qquad \leq \ \mathbb{E}((d - d_k)_+).$$

The structure of the constraint matrix is sloppily pictured in Figure 1. Here stochastic matrix entries are considered. Stochastic cost coefficients are also present, since the constraints $c^\top x + q^\top y_l - v_{lk} \leq d_k$ and the matrices T and W are condensed in T'' and W'', now with scenario index and hence possibly changing from scenario to scenario. We get a lower bound by solving the Lagrangean dual, which is a nonlinear concave maximization carried out with Helmberg's bundle method [12].

$$z_{\text{LD}} := \max_{\lambda \in \mathbb{R}^K_+, \ \mu \in \mathbb{R}^{M \cdot (L-1)}} \min \left\{ \begin{array}{ll} & \sum_{l=1}^{L} \pi_l \cdot g^\top x_l \\ + & \sum_{l=1}^{L} \sum_{k=1}^{K} \lambda_k (\pi_l \cdot v_{lk} - \pi_l \cdot \mathbb{E}((d - d_k)_+)) \\ + & \sum_{l=2}^{L} \sum_{m=1}^{M} \mu_{ml-1} \cdot \pi_l \cdot (x_{i_m 1} - x_{i_m l}) \quad : \\ & (x_l, y_l) \in M_l, \ \forall l \end{array} \right\} \tag{10}$$

$$= \max_{\lambda\in\mathbb{R}^K_+,\ \mu\in\mathbb{R}^{M\cdot(L-1)}} \sum_{l=1}^{L} \pi_l \min\Big\{ \begin{array}{ll} & g^\top x_l \\ + & \sum_{k=1}^K \lambda_k (v_{lk} - \mathbb{E}((d-d_k)_+)) \\ + & \sum_{m=1}^M \nu_{ml} \cdot x_{i_m l} \quad : \\ & (x_l, y_l) \in M_l \end{array} \Big\} \tag{11}$$

where

$$\nu_{ml} := \begin{cases} \sum_{l'=2}^{L} \mu_{ml'-1} \cdot \frac{\pi_{l'}}{\pi_1} & \text{, for } l = 1 \\ -\mu_{ml-1} & \text{, for } l \geq 2 \end{cases}$$

The number M of first-stage variables whose nonanticipativity constraints are treated with Lagrangean Relaxation has to be specified after the identifier RELNA in the specification file. In the current node the $M \cdot (L-1)$ nonanticipativity constraints of the M variables x_{i_m} with the biggest dispersion in the father of the current node are treated with Lagrangean Relaxation.

Another rather simple method to obtain a lower bound is to solve each subproblem with all multipliers equal to zero and to pick the maximal objective value of the subproblems as a lower bound. This works since all subproblems are relaxations of the full problem.

Upper bounds on the optimal value can be obtained by heuristics based on the solutions for the subproblems from the lower bounding procedure. Let $\bar{x}$ a "promising" candidate, received by some heuristic, cf. Section 4.4. We check the feasibility of $\bar{x}$ and "push down" the s_{ik} simultaneously (to fulfill (12)) with the L auxiliary problems:

$$\min\Big\{ \sum_{k=1}^{K} s_{ik} : \begin{array}{lll} c^\top \bar{x} + q^\top y_i - s_{ik} & \leq & d_k \\ T\bar{x} + W y_i & = & z_i \\ y_i \in Y, & s_{ik} \geq 0,\ k = 1, \ldots, K & \end{array} \Big\}$$

If they are feasible for all $i = 1, \ldots, L$, we still have to check whether

$$\sum_{i=1}^{S} \pi_i \cdot s_{ik} \leq \mathbb{E}((d-d_k)_+) \quad \forall k = 1, \ldots, K \tag{12}$$

If these constraints are fulfilled, we compute the upper bound $g^T \bar{x}$.

We elaborate a branch-and-bound algorithm that uses the described bounding procedures and successively reestablishes the equality of the components of the first-stage vector. By $\mathbf{P}$ we denote a list of problems, and $\varphi_{LB}(P)$ is a lower bound for the optimal value of $P \in \mathbf{P}$. Moreover, $\bar{\varphi}$ denotes the currently best upper bound to the optimal value of (7), and $X(P)$ is the element in the partition of X belonging to P.

Algorithm 2.1

STEP 1 (INITIALIZATION):
Let $\mathbf{P} := \{(7)\}$ *and* $\bar{\varphi} := +\infty$.

STEP 2 (TERMINATION):
If $\mathbf{P} = \emptyset$ *then the* $\bar{x}$ *that yielded* $\bar{\varphi} = g^\top \bar{x}$ *is optimal.*

STEP 3 (BOUNDING):
Select and delete a problem P *from* $\mathbf{P}$*. Compute a lower bound* $\varphi_{LB}(P)$ *and apply a feasibility heuristics to find a feasible point* $\bar{x}$ *of* P*.*

STEP 4 (PRUNING):
If $\varphi_{LB}(P) = +\infty$ *(infeasibility of a subproblem)) or* $\varphi_{LB}(P) > \bar{\varphi}$ *(inferiority of* P*), then go to Step 2.*
If $\varphi_{LB}(P) = g^\top \bar{x}$ *(optimality for* P*), then check whether* $g^\top \bar{x} < \bar{\varphi}$*. If yes, then* $\bar{\varphi} := g^\top \bar{x}$*. Go to Step 2.*
If $g^\top \bar{x} < \bar{\varphi}$*, then* $\bar{\varphi} := g^\top \bar{x}$*.*

STEP 5 (BRANCHING):
Create two new subproblems by partitioning the set $X(P)$ *by means of linear inequalities. Add these subproblems to* $\mathbf{P}$ *and go to Step 2.*

3 Input files

`ddsip.vSD` requires 4 input files, a number of other files are optional. The file-formats will be described in the following sections.

Mandatory:

- Specification file: a file containing the specifications of the stochastic program, some `CPLEX`- and Branch&Bound-parameters
- Model file: a mps-file readable by `CPLEX` that specifies the single-scenario model
- Scenario file: a file containing the scenario-probabilities and stochastic right-hand sides if there are any
- A Reference file: a file containing the specifications of the benchmark random variable d

Optional:

- Priority order file for subproblems: a `CPLEX` order file ([14]) corresponding to the model file
- Matrix scenario file: a file containing stochastic matrix entries
- Priority order file for master: a file containing a branching order for the master branch-and-bound procedure
- Start information file: a file containing start informations such as a feasible solution or a lower bound

A comfortable way to invoke the program on a Unix system is the command

ddsip.vSD < files2sip

where the file *files2sip* may contain the lines displayed in Figure 2.

```
sip.in
model.mps
model.ord
rhs.sc
matrix.sc
order.dat
start.in
```

Figure 2: Input file

4 The specification file

4.1 Parameters for the dominance-constrained model

A sample specification file is given in Appendix A. Hopefully, the identifiers in the first part of this file are self explaining. Otherwise, they should become clear when consulting the two-stage chapter of one of the standard text books on stochastic programming, see [3, 16, 23]. Some of the identifiers, as e.g. FIRSTVAR, are redundant. They serve to check consistency with the model file.

4.2 `CPLEX` parameters

The `CPLEX` parameters have to be specified behind the indicator *CPLEXBEGIN*. The lines have to start with the `CPLEX` parameter number followed by the parameter value, cf. Appendix A. The parameter number corresponds to the values defined in the `CPLEX` callable library. For the evaluation of upper bounds the parameter values can be overwritten after the identifier *CPLEXUB*. The `CPLEX` parameter section has to end with the identifier *CPLEXEND*. Parameters not present in this chapter are set according to their `CPLEX` default values, cf. [14].

4.3 Parameters for the decomposition procedure

The parameters that effect the amount of output and the termination behavior are listed in the table below. The first column of the table contains the name of the parameter, the second column specifies whether it is an integer or a real (Dbl) parameter. Then the parameter's range and it's default value is given. The last column explains the parameter's meaning.

Name	Type	Range	Default	Description
OUTLEV	Int	0..30	0	Print output to *more.out*. **Caution:** The file *more.out* may become large for high values of OUTLEV!
OUTFIL	Int	0..9	0	Print output files, see chapter 8. **Caution:** If OUTFIL is greater than 8, lp-files are written at each node and for each scenario!
LOGFRE	Int	0..	1	A line of output is printed every i-th iteration.
NODELI	Int	0..	1	The node limit for the branch-and-bound procedure.
TIMELI	Dbl	0..	100	The total time limit (CPU-time)
ABSOLU	Dbl	0..	0	The absolute duality gap. (Should be greater than RELATI)
RELATI	Dbl	0..	0	The relative duality gap.
OBJISINT	Int	0..1	0	Indicates whether the objective function attains only integer values. If yes lower bounds are rounded up.
DOMINANCE	Int	1..2	–	Indicates which if first- or second order stochastic dominance is used
DETEQU	Int	0..1	0	Indicates whether a deterministic equivalent to the current problem is written out to *sipout*
TEST	Int	0..1	0	Indicates whether a deterministic equivalent with fixed variables is written out after a problem was solved
TRIVIALUB	Dbl	..	∞	Value that is interpreted as infinity by the dual method (dual method in the current node stops when lower bound exceeds TRIVIALUB and the lower bound is set to infinity)
SCALEREF	Dbl	..	0	SCALEREF is added to every outcome of d. It can be made easier or harder to be fulfilled by this value

Table 1: Output and termination parameters

So far all termination parameters are static in the sense that they cannot be changed during the branch-and-bound algorithm. Therefore, a careful trade-off between these parameters and the termination parameters of the bundle method, see Section 4.5, is essential for the

numerical performance, cf. [6].

Parameters that effect the behavior of the branch-and-bound algorithm are compiled in Table 2. When confusion is possible, the default values are marked with a star. The use of start values is described in Section 7.1, the use of priority order information in Section 7.2.
The dispersion norm of a node is calculated as $\max_j\{\max_i x_{ij} - \min_i x_{ij}\}$ where x_{ij} denotes the i-th component of the firststage part of the solution of the j-th scenario. Nodes with a dispersion norm smaller than NULLDI are considered as leaves of the complete enumeration tree.

Name	Type	Range	Default/Description	
PORDER	Int	0..1	0	Use priority order for branching? If this parameter is 1, then the priority order file needs to be specified.
STARTI	Int	0..1	0	Use start information? If this parameter is 1, a file containing the start values needs to be specified.
BRADIR	Int	-1,1	-1	Branching down is preferred.
			1*	Branching up is preferred.
BOUSTR	Int	0..2	0*	In order to improve bounds, the node with the worst lower bound is chosen.
			1	Width first strategy. Node with smallest index is chosen.
			2	Depth first strategy. Node with largest father index is chosen.
BRASTR	Int	0..1	0	Use the average of lower and upper bound in father node as branching value.
			1*	Use the weighted average of the scenario solutions of the component as branching value.
INTFIR	Int	0..1	1	Branch first on integer variables (despite of smaller DISPNORM)
EPSILO	Dbl	0..	0	Specifies disjoint subdomains if continuous components are branched.
NULLDI	Dbl	-1..	0	Branch nodes only if their dispersion norm is greater than NULLDI.
VIS	Int	0..2	1	Visualization of the Branch&Bound tree of the subdivision of X: 0: no visualization 1: pdf-file sipout/bbtree.pdf is written just before termination 2: visualization is done "on-the-fly". This might be very time-consuming, espe-

To be continued on the next page

Name	Type	Range	Default/Description
			cially, when the nodes are solved very quickly. It is interesting to watch. Use: "gv -watch bbtree.ps" in the directory sipout during the computation. Note that it is necessary to have the "Graphviz-package" installed. See [15] for more information. Yellow nodes: multiple Red nodes: cutoff Green nodes: otimallity
NLB	Int	..	0 Specifies the use of the simple lower bounding procedure (take maximum of the subsolutions; all Lagrangean multipliers equal to 0) described before: ..-3: use the simple procedure and CB only in every $(-i)$-th node, where i is the specified number -2: use only the simple procedure -1: use the simple procedure only in node 0 and CB in all nodes 0 : use only CB 1..: use CB and the simple procedure in every $i-$th node, where i is the specified number

Table 2: Branch-and-bound parameters

4.4 Heuristics

The decomposition algorithm uses a heuristic to guess feasible first-stage solutions based on the solutions of the subproblems in the current node. The heuristic is set by means of the parameter HEURIS. The possible values of HEURIS are listed in the following table.

Value	Description
1	The average of the subsolutions is used. Integer components are rounded down.
2	The average of the subsolutions is used. Integer components are rounded up.
3*	The average of the subsolutions is used. Integer components are rounded to the next integer.
4	The subsolution occurring most frequently is used.

To be continued on the next page

Value	Description
5	The solution of a subscenario that is closest (l_1 norm) to average is used.
6	Apply heuristic 3 and 5 alternating
7	Solution with best objective value
8	Solution with worst objective value
9	Solution with minimal sum of first-stage variables
11	Solution with maximal sum of first-stage variables
12	Try all subsolutions
13	Solve a randomly selected scenario with *high* precision, the remaining ones with *low*
14	Use solution of the first scenario.
99	Special heuristic (sipheurchanged.c has to be changed and recompiled for special requirements)

Table 3: Values of parameter HEURIS

4.5 Parameters for the dual method

Name	Type	Range	Default	Description
CBITLI	Int	0..	∞	Iteration limit: descent steps.
CBTOTI	Int	0..	∞	Iteration limit: descent and null steps.
CBPREC	Dbl	0..	1e-7	Precision of bundle method.
INITMULT	String			Path of file with multipliers initially used by Conic Bundle
RELNA	String			Number of first-stage variables whose nonanticipativity constraints are treated with Lagrangean Relaxation

Table 4: `ConicBundle` parameters

5 The model file

The model file contains the single-scenario model in the mps-format ([14]).

- The sense and names of the first-stage constraints (constraints only involving first-stage variables) have to be placed after "N obj" (the objective function) at the beginning of the ROWS-section of the mps-file.

- The names of the first-stage constraints have to be followed by the names of the second-stage constraint with a stochastic right-hand-side.
- Then all "other" second stage constraints follow.
- At the end of the ROWS-section the name of the constraint $c^\top x + q^\top y - v \leq d$ has to be put. Here d is an arbitrary value, since it is adjusted by the decomposition method.
- The first-stage variables have to be placed at the beginning of the COLUMNS-section of the mps-file and their names have to end with the identifier '01'.
- They are followed by all second stage variables.
- The last variable in the COLUMNS-section has to be a nonnegative continuous variable which is interpreted as the variable v.
- For **first-order stochastic dominance** the constraint $c^\top x + q^\top y - v \leq d$ has to be modified according to $c^\top x + q^\top y - M \cdot \theta \leq d$. M has to be appropriately chosen by the user and the last variable in the COLUMNS-section has to be a binary variable instead of a nonnegative continuous variable (cf. [11]).

Below there is a small example model-file with 1 first-stage constraint ($\geq$), 2 first-stage variables (2 nonnegative continuous variables), 3 second-stage constraints (($\geq, \leq, \leq$) 1 with stochastic right-hand-side and 1 with stochastic matrix entries) and 3 second-stage variables (1 binary variable and 2 nonnegative continuous variables). We recommend to built up the model-file with some modelling-language interpreter as ILOG OPL-Studio (free trial on [13]) or the free software Zimpl by Thorsten Koch [26]. To postprocess the mps-file pick a text-editor of your choice. For instance "vi" has nice functions to easily achieve the special format, that is needed by our software and can be found on any Unix-/Linux-distribution.

- Note that the constraint $\sum \pi_i \cdot v_{ik} \leq \ldots$ does not occur in the model. It is added during the algorithm.
- The constraint $c^\top x + q^\top y - v \leq d$ has to occur only once, independently on the number of reference outcomes.

```
NAME
ROWS
 N  obj
 G  FirCon
 G  StocRhsSecCon
 L  StocMatSecCon
 L  cx_qy
COLUMNS
    x1_01          obj                       1
    x1_01          FirCon                    1
    x1_01          StocMatSecCon             1
    x1_01          cx_qy                     1
    x2_01          FirCon                    1
    MARK0000  'MARKER'                 'INTORG'
```

```
    y             StocRhsSecCon            1
    y             StocMatSecCon            1
    y             cx_qy                    1
    MARK0001  'MARKER'                 'INTEND'
    z             StocRhsSecCon            1
    z             StocMatSecCon            1
    v             cx_qy                   -1
RHS
    rhs           FirCon                   2
    rhs           StocRhsSecCon            1
    rhs           StocMatSecCon           30
    rhs           cx_qy                  100
BOUNDS
 UP bnd           y                        1
ENDATA
```

6 The scenario files and the benchmark file

Figure 3 displays the different scenario files.

right-hand sides	matrix entries	benchmark distribution	
scenario1	position		
0.1	200	0.05	1
23	186	0.15	3
34		0.25	4
⋮	⋮	⋮	⋮
scenarioL	scenario1	0.1	15
0.1	23		
30	34		
41			
⋮	⋮		
	scenarioL		
	30		
	41		

Figure 3: Format of the scenario files

In the right-hand side scenario file, the first number after the identifier *sce* is the scenario probability. For problems without stochastic right-hand sides, this file contains only the probabilities of the individual scenarios. In order to assign the stochastic right-hand sides to constraints, the constraints have to be grouped in the model file, cf. Section 5.
The matrix scenario file requires additionally a row and column index for each stochastic matrix coefficient, see Figure 3. The identifier for this index is *pos*. If r is the number of stochastic matrix entries, the $(2i-1)$-th number $(1 \leq i \leq r)$ following *pos* is treated as row index and the $(2i)$-th number as column index of the i-th entry of the individual scenarios.

The program "siphelp“ makes things a lot easier and can be obtained from our workgroup. The file containing the information what the benchmark distribution looks like plays a prominent role. It should consist of two columns.

- First Column: contains the probabilities that d attains d_k for $k = 1, \ldots, K$
- Second Column: contains the corresponding realizations d_k for $k = 1, \ldots, K$

These pairs do not have to be sorted, but the probabilities should sum up to 1.

7 Optional files

7.1 Start information file

Start information can be provided as displayed in Figure 4. Hereby, BEST should be an upper

```
BEST        2345
BOUND       2000
SOLUTION
1
2
⋮
```

Figure 4: File with start values

bound, BOUND a lower bound and SOLUTION a feasible first-stage solution. The user has to care about the consistency of the data.

7.2 The priority order file

The priority order for the branching in the master problem is specified by passing a file to `ddsip.vSD`. The file has two columns where the first contains indices of first-stage variables and the second integer values. High values lead to early branching.

7.3 The multiplier file

The multiplier file has one column with as many nonnegative values as the benchmark-profile has realizations. It specifies the initial multipliers, that Conic Bundle uses. To use a multiplier-file the parameter INITMULT has to occur in the specification file, followed by the path of the file.

8 Output

8.1 Output on screen

Figure 5 shows two typical lines of output as produced by `ddsip.vSD`. Hopefully, the meaning of the single entries is rather straight forward. Here is a short explanation.

Nodes	Left	Objective	Strategy	Heuristic	Best Value	Bound	Gap	Time
11	5	25.2384	(CB)	25.9449	25.8430	24.9897	3.3%	0.25

Figure 5: Output lines

Nodes	counts the number of nodes generated in the tree.
Left	counts the number of nodes in the front tree.
Objective	is the lower bound of the current node.
Strategy	is the information which lower bounding strategy was used
Heuristic	is the upper bound returned by the heuristic.
Best Value	is the overall upper bound.
Bound	is the overall lower bound.
Gap	is the relative gap between *Best Value* and *Bound*.
Time	is the CPU time passed since invoking the program.

The pruning of nodes is indicated by the entry *cutoff* in the row 'Objective'. The row 'Heuristic' may also contain the entries *n-stop* (inferiority, evaluation stopped at n-th scenario) or *multiple* (a solution has been evaluated previously).

8.2 Output in the file *sip.out*

All output files will be placed in the subdirectories *sipout* and *debug* of the current directory. These directories will be created if they do not exist. A further run of `ddsip.vSD` overwrites the existing output files. The output on screen is also directed to the file *sip.out*. In addition, this file contains the read parameters and some information on the solution:

- The value of *Status* indicates the solution status:
  - -1 the process was terminated by the user,
  - 1 the node limit has been reached,
  - 2 the gap (absolute or relative) has been reached,
  - 3 the time limit has been reached,
  - 4 the maximal dispersion, i.e. the maximal difference of the first stage components within all remaining front nodes, was less then the parameter NULLDISP (null dispersion),
  - 5 the whole branching tree was backtracked.
- *Time* is the total time needed by `ddsip.vSD`.
- *Upper bounds* is the number of evaluated upper bounds.
- *Tree depth* is the depth of the branch-and-bound tree.
- *Nodes* is the total number of nodes.

8.3 Other output files

The number of additional output files and their content is ruled via the parameters OUTFIL and OUTLEV.

- The files *rhs.out*, *matrix.out*, and *model.mps* offer a check for a correct reading of the scenario files and the model file, respectively.
- The files *solution.out* contains information on the solution with the optimal first-stage solution among them.
- The file *more.out* contains more or less information depending on the parameter OUTLEV, e.g. the subproblem solutions, the branch-and-bound tree, and the objective function composition.
- *det_equ_FSD/SSD.mps.gz, test.lp.gz:* Deterministic equivalent of the current problem and deterministic equivalent with fixed variables, if the decomposition algorithm exited normally and if DETEQU/TEST were set to 1.

9 Licence and bug

9.1 Licence

The program is free software. It is distributed under the GNU General Public License as published by the Free Software Foundation, see [9].

9.2 Bug report

Please report bugs via email to gotzes@math.uni-duisburg.de and neise@math.uni-duisburg.de. If possible, include the input files to make the error reproducible.

A Sample specs file

```
* Parameters to specify the model

FIRSTCON          9   * First stage constraints
FIRSTVAR         30   * First stage variables
SECCON          280   * Second stage constraints
SECVAR          326   * Second stage variables
SCENARIOS         5   * Number of scenarios
STOCRHS          70   * Number of stochastic rhs elements
STOCMAT           0   * Number of stochastic matrix entries

* Parameters for dual decomposition procedure

OUTLEVEL          5   * Print info to more.out
OUTFILES          1   * Print files (models and output)
STARTINFO         0   * Use start solution/bounds
NODELIM          90   * Node limit for the outer b&b
TIMELIM         800   * ddsip.vSD time limit
HEURISTIC         4   * Heuristic to produce a feasible solution
ABSOLUTE          0   * Absolute duality gap
RELATIVE      0.001   * Relative duality gap

PORDER            0   * Use branching priority order
BRADIR           -1   * Branching direction
BRASTRAT          0   * Branching strategy
BOUSTRAT          1   * Bounding strategy
EPSILON        1e-4   * Epsilon used for branching on real variables
LOGFREQ           1   * Output log frequency
INTFIRST          1   * Branch on integers first
ACCURACY       1e-8   * Accuracy used to compare real values
NULLDISP          0   * Tolerance level for null-dispersion (used together
                        with ACCURACY!)
QUANTILES        10   * Number of Quantiles printed out at the end
NLB              10   * Node-interval for "simple lower bound procedure“
VIS               1   * Visualization of the b&b-procedure
TEST              0   * Write out deterministic equivalent with fixed
                        variables
DETEQU            0   * Write out the deterministic equivalent
DOMINANCE         2   * Dominance model
REFERENCE  test.ref   * Path of the file that specifies the benchmark
SCALEREF          0   * SCALEREF is added to every outcome of d
                        It can be made easier or harder to be fulfilled
                        by this value
MREFSCALE         1   * every outcome of d is multiplied by MREFSCALE
                        It can be made easier or harder to be fulfilled
                        by this value
```

```
INITMULT        mult    * Path of the file that specifies initial multipliers
TRIVIALUB       10000   * Value that is interpreted as infinity by the
                          dual method (dual method in the
                          current node stops when lower
                          bound exceeds TRIVIALUB and the
                          lower bound is set to infinity)
OBJISINT        0       * If the objective-value is integral, the dual method
                          rounds its solutions up to the next integer
RELNA           0       * Number of variables that are treated with
                          Lagrangean Relaxation


* CPLEX Parameters (See CPLEX manual [14])

CPLEXBEGIN
1035            0       * Output on screen indicator
2009            0.31    * Relative gap
CPLEXUB
1035            1       * Output on screen indicator for upper bounds
CPLEXEND

* Parameters specifying the use of ConicBundle

CBITLIM         20      * Limit for number of descent steps in Conic Bundle
CBTOTIT         1000    * Limit for the total number of Conic Bundle iterations
                          (incl. nullsteps)
```

Specification file

References

[1] Ahmed, S.: Convexity and decomposition of mean-risk stochastic programs, Mathematical Programming 106 (2006), 433–446.

[2] Bank, B.; Mandel, R.: Parametric Integer Optimization, Akademie-Verlag, Berlin, 1988.

[3] Birge, J.R.; Louveaux, F.V.: Introduction to Stochastic Programming, Springer, New York, 2000

[4] Blair, C.E.; Jeroslow, R.G.: The value function of a mixed integer program: I. Discrete Mathematics 19 (1977), 121–138.

[5] Carøe, C.C.; Schultz, R.: Dual Decomposition in Stochastic Integer Programming, Operations Research Letters 24, pp. 37-45, 1999

[6] Carøe, C.C.; Schultz, R.: A two-stage stochastic program for unit commitment under uncertainty in a hydro-thermal power system, Konrad-Zuse-Zentrum für Informationstechnik Berlin, Preprint SC 98-11, 1998

[7] Drapkin, D.; Schultz, R.: A Decomposition Algorithm for Stochastic Programs with First-Order Dominance Constraints Induced by Linear Recourse, Preprint 653–2007, Department of Mathematics, University of Duisburg-essen, 2007

[8] Eichhorn, A.; Römisch, W.: Polyhedral risk measures in stochastic programming, SIAM Journal on Optimization 16 (2005), pp. 69–95.

[9] GNU Project, http://www.gnu.org.

[10] Gollmer, R.; Gotzes, U.; Schultz, R.: Second-Order Stochastic Dominance Constraints Induced by Mixed-Integer Linear Recourse, Preprint 644–2007, Department of Mathematics, University of Duisburg-essen, 2007.

[11] Gollmer, R.; Neise, F.; Schultz, R.: Stochastic Programs with First-Order Dominance Constraints Induced by Mixed-Integer Linear Recourse, Preprint 641–2006, Department of Mathematics, University of Duisburg-essen, 2006.

[12] Helmberg, C.: http://www-user.tu-chemnitz.de/~helmberg/Conic Bundle/Manual/ , 2005

[13] http://www.ilog.com/products/optimization/

[14] ILOG CPLEX 9.0 Manuals, http://www.iro.umontreal.ca/~gendron /IFT6551/CPLEX/HTML/.

[15] http://www.graphviz.org/

[16] Kall, P.; Wallace, S.W.: Stochastic Programming, Wiley, Chichester, 1994

[17] Louveaux, F.V.; Schultz, R.: Stochastic Integer Programming, In: A. Ruszczynski, A. Shapiro (Eds.), Stochastic Programming, Elsevier, North-Holland, 2003

[18] Müller, A.; Scarsini, M.: Stochastic Order Relations and Lattices of Probability Measures, Siam Journal on Optimization, 16 (2006), pp. 1024-1043

[19] Märkert, A.: Deviation Measures in Stochastic Programming with Mixed-Integer Recourse, Doctoral thesis, Institute of Mathematics, University Duisburg-Essen, Campus Duisburg, 2004

[20] Märkert, A.: ddsip: http://www.uni-duisburg.de/FB11/disma /ddsip.shtml

[21] Mann, H.B.; Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other, Annals of Mathematical Statistics 18 (1947), 50–60.

[22] Müller, A.; Stoyan, D.: Comparison Methods for Stochastic Models and Risks, Wiley, Chichester, 2002.

[23] Prekopa, A.: Stochastic Programming, Kluwer, Dordrecht, 1995

[24] Schultz, R.; Tiedemann, S.: Conditional value-at-risk in stochastic programs with mixed-integer recourse, Mathematical Programming 105 (2006), pp. 365–386.

[25] Schultz, R.; Tiedemann, S.: Risk Aversion via Excess Probabilities in Stochastic Programs with Mixed-Integer Recourse, SIAM Journal on Optimization 14, (2003), pp. 115-138

[26] http://www.zib.de/koch/zimpl/